LLMs as Zero-shot Graph Learners: Alignment of GNN Representations with LLM Token Embeddings

Duo Wang **Yuan Zuo*** **Fengzhi Li** **Junjie Wu**
MIIT Key Laboratory of Data Intelligence and Management, Beihang University
{wangduo58, zuoyuan, lifengzhi, wujj}@buaa.edu.cn

Abstract

Zero-shot graph machine learning, especially with graph neural networks (GNNs), has garnered significant interest due to the challenge of scarce labeled data. While methods like self-supervised learning and graph prompt learning have been extensively explored, they often rely on fine-tuning with task-specific labels, limiting their effectiveness in zero-shot scenarios. Inspired by the zero-shot capabilities of instruction-fine-tuned large language models (LLMs), we introduce a novel framework named Token Embedding-Aligned Graph Language Model (TEA-GLM) that leverages LLMs as cross-dataset and cross-task zero-shot learners for graph machine learning. Concretely, we pretrain a GNN, aligning its representations with token embeddings of an LLM. We then train a linear projector that transforms the GNN's representations into a fixed number of graph token embeddings without tuning the LLM. A unified instruction is designed for various graph tasks at different levels, such as node classification (node-level) and link prediction (edge-level). These design choices collectively enhance our method's effectiveness in zero-shot learning, setting it apart from existing methods. Experiments show that our graph token embeddings help the LLM predictor achieve state-of-the-art performance on unseen datasets and tasks compared to other methods using LLMs as predictors. Our code is available at `https://github.com/W-rudder/TEA-GLM`.

1 Introduction

Graph Neural Networks (GNNs) have emerged as a pivotal framework in graph machine learning, harnessing the ability to capture intricate message-passing patterns for robust graph representation. These advancements have yielded various GNN architectures, including the Graph Convolution Network (GCN) [1], Graph Attention Network (GAT) [2], and GraphSAGE [3]. Despite their efficacy, GNNs often exhibit limited generalization capabilities, struggling to maintain consistent performance when transitioning across different datasets or downstream tasks [4]. This limitation underscores the necessity for more adaptable and universally applicable models in the graph learning domain.

To mitigate the dependency on labeled data and enhance the resilience of graph models, self-supervised learning has been widely adopted in GNN training. Techniques such as Deep Graph Infomax (DGI) [5] and GraphCL [6] have demonstrated effectiveness by leveraging mutual information maximization and contrastive learning, respectively. However, these methods typically require fine-tuning task-specific heads for downstream applications, which can be resource-intensive and limit their practicality in diverse scenarios. Moreover, graph prompt learning enhances GNN generalization by using unified task templates and meta-learning to adapt to various downstream applications [7, 8], but it often requires extensive fine-tuning and is constrained by the specificity of task types.

In recent years, the remarkable generalization capabilities of Large Language Models (LLMs) have spurred interest in their potential applications within graph machine learning. Some methods attempt

*Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

to encode graph structures into text for LLM input [9, 10, 11, 12], but these approaches often lead to suboptimal outcomes [13]. Alternatively, using LLMs as enhancers to generate data or node text representations [14, 15, 16, 17, 18] has shown promise but remains constrained by the inherent reliance on GNNs for prediction. Recent efforts [19, 20] to use LLMs as predictors have demonstrated potential. However, their performance often remains unstable due to the challenge of producing transferable graph representations that work effectively for LLMs across diverse tasks and datasets.

In light of these challenges, we propose a novel framework named Token Embedding-Aligned Graph Language Model (TEA-GLM). Inspired by the zero-shot capabilities of instruction-fine-tuned LLMs [21], TEA-GLM leverages LLMs as cross-dataset and cross-task zero-shot predictors for graph machine learning. The core idea is to pretrain a GNN and align its representations with the token embeddings of an LLM. This alignment enables the GNN to effectively utilize the LLM's pretrained knowledge, allowing it to generalize across different datasets and tasks without task-specific fine-tuning. Additionally, we train a linear projector to convert graph representations into a fixed number of token embeddings, which are then incorporated into a unified instruction designed for various graph tasks at different levels. Experiments show TEA-GLM achieves superior performance in zero-shot scenarios and when encountering unseen tasks, offering a more generalized and efficient solution for graph zero-shot learning. Our contributions are summarized as follows:

- We introduce TEA-GLM, a novel framework that aligns GNN representations with LLM token embeddings, enabling cross-dataset and cross-task zero-shot learning for graph machine learning.
- We propose a linear projector that maps graph representations into a fixed number of graph token embeddings. These embeddings are incorporated into a unified instruction designed for various graph tasks at different levels, enhancing the model's generalization capabilities.
- Our extensive experiments demonstrate that TEA-GLM significantly outperforms state-of-the-art methods on unseen datasets and tasks.

2 Methodology

In this section, we introduce TEA-GLM, a novel framework designed for cross-dataset and cross-task zero-shot graph machine learning. TEA-GLM consists of two main components: a Graph Neural Network (GNN) to derive node representations from the graph, and a Large Language Model (LLM) to perform zero-shot tasks such as node classification and link prediction. Our methodology involves two key stages: enhanced self-supervised learning of the GNN, where feature-wise contrastive learning with LLM's token embeddings is proposed, and training a linear projector to map graph representations into a fixed number of graph token embeddings by designing an instruction that is suitable for various graph tasks at different levels. The framework of our proposed method is illustrated in Fig. 1.

2.1 Notations

Formally, a graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A}, \mathbf{X})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ with $|\mathcal{V}| = N$ indicating the total number of nodes and $\mathcal{E} = \{e_1, e_2, \ldots, e_{|\mathcal{E}|}\}$ representing the sets of nodes and edges, respectively. The adjacency matrix is denoted as $\mathbf{A} \in \mathbb{R}^{N \times N}$, with $\mathbf{A}_{ij} = 1$ iff $(v_i, v_j) \in \mathcal{E}$. The feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F_N}$ contains the attribute or feature information associated with each node, where $\boldsymbol{x_i} \in \mathbb{R}^{F_N}$ is the feature of v_i, and F_N represents the dimensionality of features.

2.2 Token embeddings-aligned graph self-supervised learning

Given the increasing model sizes and data volumes in recent years, self-supervised learning has become a prominent research focus due to the scarcity of labeled data. In this context, we propose a contrastive learning method to obtain more transferable node representations suitable for use with large language models (LLMs). Our approach leverages instance-wise contrastive learning and introduces a feature-wise contrastive learning method that maps node representations to the textual embedding space of the LLM.

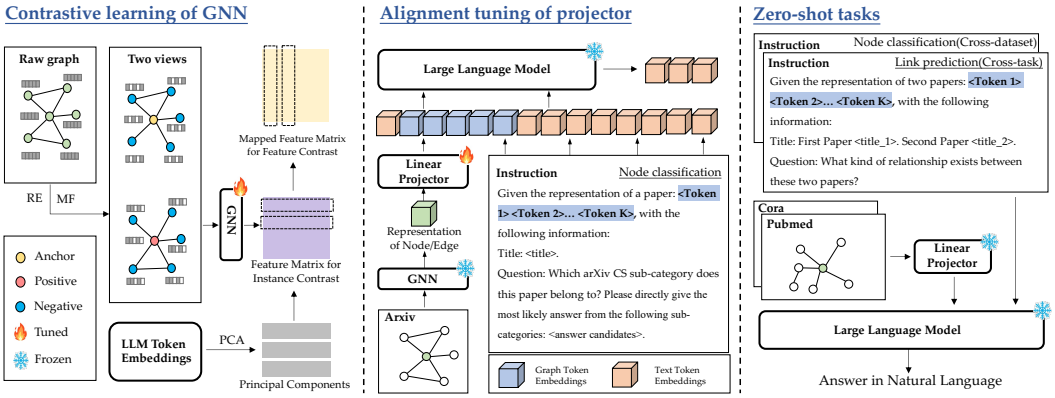


Figure 1: Framework of TEA-GLM

2.2.1 Instance-wise contrastive learning with structural information

To alleviate the need for labeled data and enhance model generalization capability, we employ self-supervised learning for pre-training. To better extract structural information from the graph, we follow the work of [22] to generate two views of $\mathcal{G}$, denoted as $\mathcal{G}_1$ and $\mathcal{G}_2$, for contrastive learning. Specifically, we adopt the Removing Edges (RE) and Masking Node Features (MF) methods to generate different views. The RE strategy samples a random masking matrix $\widetilde{\mathbf{R}} \in \{0,1\}^{N\times N}$ to mask the raw adjacency matrix, computed as:

$$\widetilde{\mathbf{A}} = \mathbf{A} \circ \widetilde{\mathbf{R}}, \tag{1}$$

where $\circ$ denotes the Hadamard product. The MF strategy samples a random mask vector $\widetilde{\boldsymbol{m}} \in \{0,1\}^{F}$. The generated node features $\widetilde{\mathbf{X}}$ are computed by:

$$\widetilde{\mathbf{X}} = \left[\boldsymbol{x_1} \circ \widetilde{\boldsymbol{m}}; \boldsymbol{x_2} \circ \widetilde{\boldsymbol{m}}; \ldots; \boldsymbol{x_N} \circ \widetilde{\boldsymbol{m}}\right]. \tag{2}$$

Thus, we obtain two views of $\mathcal{G}$, denoted as $\mathcal{G}_1 = \left(\widetilde{\mathbf{X}}_1, \widetilde{\mathbf{A}}_1\right)$ and $\mathcal{G}_2 = \left(\widetilde{\mathbf{X}}_2, \widetilde{\mathbf{A}}_2\right)$. Then, we use a graph encoder to derive node representations:

$$\mathbf{U}_* = f_{\mathrm{GNN}}\left(\widetilde{\mathbf{X}}_*, \widetilde{\mathbf{A}}_*\right) \in \mathbb{R}^{N\times F_U}, \tag{3}$$

Where F_U is the dimension size of node representations. Here, $* \in \{1,2\}$ represents different views of the graph.

We employ a contrastive objective to distinguish the embeddings of the same node in these two different views from other node embeddings. For node v_i, its node embedding generated in one view, $\boldsymbol{u_i}$, is treated as the anchor, while the embedding generated in the other view, $\boldsymbol{u_i}'$, forms the positive sample. Embeddings of other nodes in the same view are regarded as intra-view negative samples, while embeddings of other nodes in the other view are regarded as inter-view negative samples. The contrastive loss is defined as:

$$\ell\left(\boldsymbol{u_i}, \boldsymbol{u_i}'\right) = \log \frac{e^{\theta\left(\boldsymbol{u_i}, \boldsymbol{u_i}'\right)/\tau}}{\underbrace{e^{\theta\left(\boldsymbol{u_i}, \boldsymbol{u_i}'\right)/\tau}}_{\text{the positive pair}} + \underbrace{\sum_{j=1}^{N} 1_{[j\neq i]} e^{\theta\left(\boldsymbol{u_i}, \boldsymbol{u_j}\right)/\tau}}_{\text{intra-view negative pairs}} + \underbrace{\sum_{j=1}^{N} 1_{[j\neq i]} e^{\theta\left(\boldsymbol{u_i}, \boldsymbol{u_j}'\right)/\tau}}_{\text{inter-view negative pairs}}}, \tag{4}$$

where $1_{[j\neq i]} \in \{0,1\}$ is an indicator function that equals 1 iff $j \neq i$, $\theta\left(\cdot,\cdot\right)$ is the cosine similarity function, and τ is a temperature parameter. The loss for the other view is similarly defined, and the overall objective $\mathcal{L}_{ins}$ is the average of all instances:

$$\mathcal{L}_{ins} = \frac{1}{2N} \sum_{i=1}^{N} \left[\ell\left(\boldsymbol{u_i}, \boldsymbol{u_i}'\right) + \ell\left(\boldsymbol{u_i}', \boldsymbol{u_i}\right)\right]. \tag{5}$$

To enhance the scalability of our method for large-scale graphs, we employ the subsampling approach proposed by [3]. Both the RE and MF methods, along with the loss function described in Equation 4, are seamlessly adaptable to the sampled subgraphs.

2.2.2 Feature-wise contrastive learning with token embeddings

Instance-wise contrastive learning relies heavily on individual instances, which can cause transfer issues when transitioning to other datasets. Moreover, there is a significant gap between the obtained node representations and the semantic space of LLMs. To address these issues, we propose feature-wise contrastive learning with token embeddings.

Feature-wise contrastive loss breaks the independence between instances. For the feature matrix $\mathbf{U}_*$, we denote the columns in different views as $\boldsymbol{m_i} \in \mathbf{U_1^\top}$ and $\boldsymbol{n_i} \in \mathbf{U_2^\top}$. Here, $\boldsymbol{m_i}, \boldsymbol{n_i} \in \mathbb{R}^N$. The loss is denoted as $\mathcal{L}_{fea}$, and is calculated as:

$$\mathcal{L}_{fea} = \frac{1}{F_U} \sum_{i=1}^{F_U} \log \frac{e^{\theta(\boldsymbol{m_i}, \boldsymbol{n_i})/\tau}}{\sum_{j=1}^{F_U} \left[e^{\theta(\boldsymbol{m_i}, \boldsymbol{m_j})/\tau} + e^{\theta(\boldsymbol{m_i}, \boldsymbol{n_j})/\tau} \right]}. \tag{6}$$

To map node representations to the semantic space of LLMs, we use the principal components of the token embeddings of LLMs as coordinate axes. This approach ensures that the representations of similar instances are closely aligned in the textual embedding space. This helps alleviate the inconsistency in optimization objectives during graph self-supervised learning due to the gap between node representations and the text embedding space.

Specifically, we first use principal component analysis (PCA) to obtain the P principal components, denoted as $\mathbf{C} \in \mathbb{R}^{P \times F_L}$, where F_L is the dimension size of token embeddings of LLM. Then, we map node representations by:

$$\widetilde{\mathbf{U}}_* = \mathbf{U}_* \times \mathbf{C}^\top. \tag{7}$$

To map the node representations obtained from the GNN using principal components, we set the output dimension of the GNN to be equal to the token embeddings' dimension (i.e., $F_U = F_L$). The columns of the mapped feature matrix $\widetilde{\mathbf{U}}_*$, denoted as $\widetilde{\boldsymbol{m}}_i$ and $\widetilde{\boldsymbol{n}}_i$, are fed into $\mathcal{L}_{fea}$. Therefore, the final contrastive loss for graph self-supervised learning is the average of Equation 4 and Equation 6:

$$\mathcal{L} = \frac{1}{2} \left(\mathcal{L}_{ins} + \mathcal{L}_{fea} \right). \tag{8}$$

Remark: The introduction of feature-wise contrastive learning with token embeddings successfully addresses the semantic space discrepancy between graph node representations and LLM token embeddings. Our method enables the direct and simple use of graph structural and text information obtained by GNN in LLMs, thereby avoiding the significant generalization loss associated with complex modality alignment training during the fine-tuning process. Its role in fine-tuning will be further described in Sec. 2.3.2 and validated by experiments. Additionally, the feature-wise contrastive method itself exhibits stronger generalization, allowing it to perform well on unseen instances (or tasks) rather than relying on trained instances (or tasks).

2.3 Alignment tuning

The development of LLMs has introduced a new paradigm for graph machine learning. However, existing research [13] indicates that LLMs alone cannot fully comprehend graph structures and their underlying information. To enable LLMs to more effectively capture information and improve their performance in cross-dataset and cross-task zero-shot learning, it is essential to design specific methods for LLMs to incorporate graph information suitably. To this end, we propose an alignment tuning method that includes specially designed instructions for various graph tasks at different levels, as well as a graph representation to graph token embeddings mechanism to integrate graph information.

2.3.1 Instructions design

The instruction we designed can be divided into two parts: one part provides graph information, and the other part describes the task. Here, we take a citation graph as an example, where nodes are papers, and relations are citations, to introduce the instruction.

Graph information provision The graph information provision in the instructions for node, edge, and graph-level tasks is presented as follows: *Given the representation of a paper/two papers/a paper set:* $\langle\mathsf{graph}\rangle$*, with the following information:\nTitle: First Paper:* $\{\mathsf{title}_1\}$ *...\n*, where $\langle\mathsf{graph}\rangle$ is the placeholder for graph inputs (see Sect. 2.3.2), and $\{\mathsf{title}_1\}$ is the node text information.

Note that, different from most work which use LLM as a predictor, the instruction we designed uses only the title of a paper node, excluding more extensive textual information such as its abstract or description. In fact, reducing the amount of input text not only does not decrease the model's performance but actually improves it. [13] confirmed through experiments that LLMs benefit from structural information only when the target node lacks sufficient phrases for reasonable predictions. Therefore, using only titles as text input can help LLMs extract more critical information from graph information. The complete instruction for the tasks of node classification and link prediction in citation networks is shown in Appendix D.

Task description To achieve cross-dataset capability, where the model can be trained on one graph dataset and then perform reasoning on any other dataset, the instruction is designed to include not only the task description itself but also the set of alternative answers. Using the node classification task on the Arxiv dataset (see Sect. 3.1) as an example, the instruction is structured as follows: *Which arXiv CS sub-category does this paper belong to? Please directly give the most likely answer from the following sub-categories:* $\{\mathsf{ans}\}$, where $\{\mathsf{ans}\}$ represents the set of alternative answers, which varies across datasets. Including alternative answers enables the model to learn the task of "reasoning the answer from a given set according to the task" rather than memorizing answers for a particular dataset, thus facilitating reasoning across datasets.

2.3.2 Graph token embeddings

The token embeddings of graph mentioned previously, *i.e.*, $\langle\mathsf{graph}\rangle$, are crucial for incorporating graph information and enabling the model's generalization. We use a projector to map central node representations into K graph token embeddings and replace $\langle\mathsf{graph}\rangle$ with these tokens. Kindly note that, we map the representations to fixed number of token embeddings regardless of the task type. For example, for node-level tasks, we map the central node representation to K token embeddings; for edge-level tasks, we pool the representations of the two nodes of the target edge and then map this pooled representation to K token embeddings; for graph-level tasks, similar approach can be applied. In this way, we unify the instruction of graph tasks at different levels. Thanks to the text-aligned contrastive learning, a linear projector is enough to capture the map relationship without tuning LLM:

$$\mathbf{H}_{token} = f_{\text{Linear}}\left(\boldsymbol{u_i}\right) \tag{9}$$

where $\boldsymbol{u_i} \in \mathbf{U}$, $\mathbf{H}_{token} \in \mathbb{R}^{K \times F_L}$, F_L is the dimension size of token embedding of LLM, and $f_{\text{Linear}}(\cdot)$ is a linear layer.

Remark: This approach offers three primary advantages: (i) When handling tasks at different levels, the changes to the instructions are minimal. This consistency facilitates the transfer of knowledge learned during training to unseen tasks in large language models (LLMs); (ii) The fixed number of token embeddings can be seen as a conditional soft prompt. Unlike traditional soft prompts, learning at the instance level reduces the risk of overfitting to specific datasets or tasks, thereby enhancing generalization to unseen datasets and tasks; (iii) Different from current work which intends to include the representations of all nodes in the subgraph, we only map the representations of the central node to tokens, since there has enough information carried by message passing of GNN. This method is more efficient, and it offers greater generalizability and practicality.

2.3.3 Training and evaluation strategy

To ensure compatibility and facilitate comparisons across various datasets, we map the node features into a consistent vector space. Specifically, we employ a pretrained BERT model [23] to encode the raw text associated with each node, thereby generating the node features. We then pretrain the graph model using contrastive learning with the loss function defined in Equation 8 on a single dataset. After pretraining, the model parameters are fixed. We utilize the pretrained model to obtain node representations and follow the instructions in Section 2.3.1 to train the linear projector on specific tasks within the same dataset. Finally, we evaluate the performance of our model on unseen datasets and tasks. Throughout all phases, the parameters of the language model remain fixed. We use GraphSAGE [3] as our graph encoder and Vicuna-7B-v1.5 [24] as the foundational language model.

3 Experimental results

In this section, comprehensive experiments are conducted to validate the effectiveness of TEA-GLM. These experiments aim to investigate the following research questions:

RQ1: How effective is TEA-GLM in handling the cross-dataset zero-shot learning problem?

RQ2: How well does TEA-GLM transfer knowledge when adapted to an unseen task and dataset in a zero-shot setting?

RQ3: What is the contribution of the feature-wise contrastive learning and graph token embeddings to the zero-shot learning ability of TEA-GLM?

3.1 Experimental setup

Datasets We test TEA-GLM across eight widely used datasets spanning two distinct domains. Within the citation domain, we employ Arxiv [25], Pubmed [26], and an expanded version of Cora [27] with an increased range of classes and larger scale. In these datasets, each node represents an individual paper, with edges indicating citation relationships. In the e-commerce domain, we utilize datasets from the TAG benchmark [28], including Children (Book-Children), History (Book-History), Computer (Ele-Computer), Photo (Ele-Photo), and Sports (Sports-Fitness). Here, nodes represent distinct products, while edges denote co-viewing or co-purchasing between two products. Appendix A presents the statistics for these datasets.

Baselines We conduct a comprehensive comparison of TEA-GLM with various categories of baseline methods: (i) Non-graph neural network approaches, such as MLP, which employs a Multilayer Perceptron for node representation; (ii) Supervised methods, including GCN [1], GraphSAGE [3], and GAT [2]; (iii) Self-supervised methods like DGI [5], which maximizes mutual information to learn node representations without relying on ground truth labels; (iv) Graph knowledge distillation frameworks: GKD [29], which distills knowledge from a teacher GNN trained on a complete graph to a student GNN operating on a smaller or sparser graph; GLNN [30], a method combining the advantages of graph neural networks and MLPs using knowledge distillation, aimed at reducing dependency on the inference graph; (v) Graph transformer networks, including NodeFormer [31] and DIFFormer [32]; (vi) Large language models, such as Vicuna-7B-v1.5; (vii) The latest models equipped with transfer and zero-shot capabilities, such as OFA [18], GraphGPT [19], and LLaGA [20].

Implementation details For datasets within the citation domain, we follow the data split methodology outlined in GraphGPT [19]. For those within the e-commerce domain, we utilize scripts provided by the TAG benchmark [28] to generate data splits. To ensure comparability among different methods, identical data splits are applied to all models. To assess the performance of TEA-GLM, we employ three commonly adopted evaluation metrics: Accuracy and Macro F1 for node classification, and AUC (Area Under the Curve) for link prediction. To ensure result robustness, we conduct five experiments with random seed values ranging from 0 to 4 and report the mean and standard deviation of the results. Due to the limited number of pages, several experimental results, such as Macro F1 results of node classification (Appendix B.2), legality rate of valid answers produced by the LLM (Appendix B.1), and parameter sensitivity analysis (Appendix C), are reported in Appendix.

In the pre-training phase of the GNN, we set the GNN layers to 2. We use a batch size of 512 for 60 epochs and a learning rate of 2×10^{-2}. During the training of the linear projector, we configure a batch size of 2 per GPU for one epoch, with a learning rate of 1×10^{-3}. The Adam optimizer is employed for all approaches. For baseline models, we adjust hyperparameters and utilize the optimal settings. All experiments are conducted on 2 NVIDIA A100 GPUs with 80GB memory each, using CUDA version 11.7.

3.2 Cross-dataset zero-shot ability (RQ1)

We train all methods on the Arxiv and Computer, respectively, followed by an evaluation of their zero-shot performance on datasets from the same domain. Zero-shot learning presents challenges for GNN-based models, particularly regarding variations in the number of classes across different datasets. To address this, we adopt the setting outlined in GraphGPT [19]. For each target dataset, we utilize the GNN backbone trained on the source dataset along with a classifier trained with target data,

Table 1: Zero-shot accuracy on citation and e-commerce datasets (**bold** highlights the best result across all methods, while underline highlights the second-best results)

Model type	Model	Citation		E-commerce			
		Pubmed	Cora	Children	History	Photo	Sports
	MLP	0.323±0.027	0.021±0.006	0.029±0.037	0.080±0.041	0.110±0.070	0.042±0.021
GNN as predictor	GCN	0.288±0.092	0.017±0.004	0.030±0.018	0.063±0.042	0.103±0.047	0.042±0.025
	GraphSAGE	0.316±0.058	0.014±0.007	0.008±0.007	0.195±0.206	0.056±0.055	0.051±0.015
	GAT	0.343±0.064	0.016±0.004	0.086±0.084	0.172±0.098	0.050±0.027	0.142±0.138
	DGI	0.329±0.103	0.026±0.009	0.082±0.035	0.218±0.168	0.224±0.127	0.049±0.017
	GKD	0.399±0.033	0.042±0.008	0.202±0.064	0.339±0.138	0.166±0.086	0.208±0.077
	GLNN	0.390±0.011	0.031±0.006	0.187±0.012	0.283±0.021	0.403±0.019	0.317±0.048
	NodeFormer	0.308±0.093	0.016±0.007	0.048±0.028	0.168±0.127	0.073±0.015	0.165±0.057
	DIFFormer	0.361±0.071	0.029±0.014	0.129±0.030	0.275±0.171	0.321±0.055	0.306±0.131
	OFA	0.314±0.059	0.130±0.019	0.064±0.086	0.052±0.049	0.340±0.026	0.101±0.071
LLM as predictor	Vicuna-7B-v1.5	0.719±0.010	0.156±0.001	0.270±0.001	0.363±0.001	0.378±0.004	0.370±0.001
	Vicuna-7B-SPT	0.768±0.036	0.168±0.018	0.227±0.015	0.281±0.088	0.350±0.061	0.230±0.018
	GraphGPT-std	0.701	0.126	-	-	-	-
	GraphGPT-cot	0.521	0.181	-	-	-	-
	LLaGA	0.793±0.036	0.168±0.032	0.199±0.007	0.146±0.067	0.276±0.069	0.352±0.033
	TEA-GLM	**0.848±0.010**	**0.202±0.014**	**0.271±0.010**	**0.528±0.058**	**0.497±0.027**	**0.404±0.010**

typically a linear layer. Due to the considerable time cost associated with training and evaluating GraphGPT on e-commerce datasets, we only report its performance on citation datasets as provided in their paper. "-std" and "-cot" denote the use of the standard procedure of dual-stage graph instruction tuning and COT instruction datasets generated by LLM, respectively. To demonstrate the difference between our work and Soft Prompt Tuning, we fine-tuned vicuna-7b-v1.5 using Soft Prompt and reported the results. The Accuracy results are presented in Table 1. As mentioned earlier, we report the Macro F1 results in Appendix B.2 and report results on two training datasets in Appendix B.3.

The results clearly demonstrate that TEA-GLM outperforms all state-of-the-art (SOTA) models, resulting in significant improvements. Comparative analysis with baseline models across all datasets highlights the robust generalization capability of TEA-GLM. Models utilizing GNN as a predictor face challenges in achieving cross-dataset transferability with traditional supervised and self-supervised learning methods. Even recently developed robust GNN-based models, such as NodeFormer, DIFFormer, and GKD, encounter similar issues. In the case of OFA, a recent framework for cross-domain learning, strong transferability is observed between topic-related datasets such as Arxiv and Cora (both related to computer science). Nevertheless, its generalization performance notably decreases on datasets with lower topic relevance, such as those in the e-commerce domain.

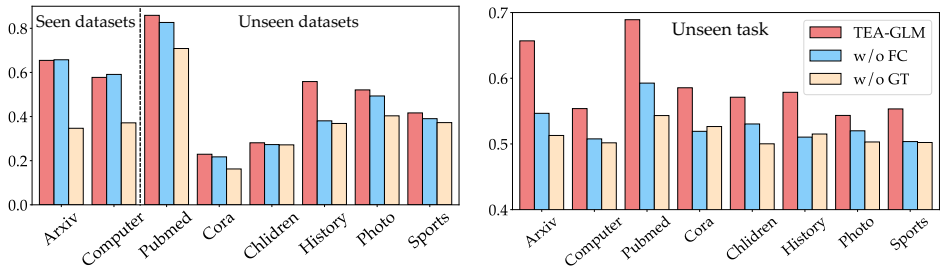
LLM-based solutions, such as Vicuna-7B, demonstrate consistent performance across various datasets. Nevertheless, their predictive capabilities are confined to text information alone. Vicuna-7B-SPT also fails to achieve transferability on e-commerce datasets, indicating that soft prompt tuning alone is insufficient when relying solely on node texts. This suggests that graph tokens indeed contain transferable graph information, enabling the LLM to make more accurate predictions. In contrast, GNN-LLM-combined solutions that use LLM as a predictor demonstrate generalization ability but often face limitations. For instance, GraphGPT tends to underperform compared to Vicuna-7B, due to the lack of a graph foundation model. Instead of relying on a graph foundation model, LLaGA directly maps node representations without GNN and can generalize on citation datasets. However, it demonstrates limited generalization capability across e-commerce datasets, which are more challenging due to highly irrelevant topics. TEA-GLM, on the other hand, utilizes principal components of token embeddings of LLMs to constrain representations learned by GNN, helping the graph representations well transfer to other datasets. Experimental results validate the superior generalization capabilities of TEA-GLM, achieved with less textual data and fewer parameters.

3.3 Cross-task zero-shot ability (RQ2)

We employ models trained on node classification tasks directly for link prediction tasks without any fine-tuning. We omit the comparison with models utilizing GNN as a predictor, as conducting cross-task evaluation of these models without fine-tuning poses a significant challenge, given that different tasks typically correspond to different task heads. Here, we contrast TEA-GLM with OFA, which similarly enables cross-task testing without the need for fine-tuning. Additionally, we compare

Table 2: AUC of link prediction (Cross-task)

Model	Citation			E-commerce				
	Arxiv	**Pubmed**	**Cora**	**Children**	**History**	**Computer**	**Photo**	**Sports**
OFA	0.469	0.481	0.492	0.484	0.431	0.461	0.459	0.517
Vicuna-7B-v1.5	0.513	0.543	0.527	0.500	0.515	0.502	0.501	0.502
Vicuna-7B-SPT	0.537	0.535	0.565	0.544	0.543	0.509	0.501	0.508
GraphGPT-std	0.649	0.501	0.520	-	-	-	-	-
LLaGA	0.570	0.569	0.537	0.422	0.449	0.479	0.478	**0.597**
TEA-GLM	**0.657**	**0.689**	**0.586**	**0.571**	**0.579**	**0.554**	**0.545**	0.553



(a) Accuracy of node classification

(b) AUC of link prediction

Figure 2: Ablation study results ("Seen datasets" are used to train the GNN and linear projector, while "unseen datasets" are not. "Unseen task" means the model wasn't trained for link prediction.)

TEA-GLM with Vicuna-7B and methods that utilize LLM as a predictor, such as GraphGPT and LLaGA. For GraphGPT, we utilize the checkpoint released by the author trained on Arxiv and report the results on citation datasets. The results are reported in Table 2.

In the case of OFA, although this framework facilitates cross-domain and cross-task learning, it exhibits negative transfer when lacking task-relevant data, particularly on unseen tasks. Benefiting from the generalization capability of large language models, both the fine-tuned and non-fine-tuned versions of Vicuna do not experience negative transfer. However, due to the absence of graph information, its predictions often appear random. Conversely, GraphGPT shows transferability with familiar datasets, yet its performance declines when dealing with unseen datasets (Pubmed and Cora). Due to the absence of GNN for filtering and aggregating graph information, LLaGA demonstrates unstable performance. While it exhibits cross-task transferability on citation datasets, its performance is poor on most e-commerce datasets. In contrast, TEA-GLM consistently outperforms all baseline methods on both unseen datasets and tasks, except for the results on Sports, indicating the stronger generalization ability of TEA-GLM.

3.4 Ablation study (RQ3)

We conduct an ablation study to discuss two key components of our model: feature-wise contrastive learning and graph token embeddings. Here, we directly remove these two components from our model and then test the model's performance on cross-dataset and cross-task evaluations. The results are shown in Figure 2. "w/o FC" means that we pretrain the GNN without feature-wise contrastive learning, while "w/o GT" means predicting without graph token embeddings.

Without graph token embeddings, large language models lack crucial information from the graph, leading to a significant decline in performance on both node-level and edge-level tasks. GNNs pre-trained with feature-wise contrastive learning can obtain node representations aligned with the text space, enabling cross-dataset and cross-task generalization through a simple linear layer. When the feature-wise constraint for pre-training is absent, the model's performance on the seen datasets (Arxiv and Computer) for the training task improves slightly. However, its performance on unseen datasets declines. Although it remains relatively stable when handling tasks of the same category, its performance decreases notably when dealing with unseen tasks (link prediction). These results

indicate that alignment between graph representation and LLM's token embeddings via feature-wise contrastive learning is important for cross-task zero-shot transfer.

4 Related work

4.1 Graph neural networks

In the field of graph machine learning, Graph Neural Networks (GNNs) have garnered significant attention [33, 34, 35, 36, 37, 38, 39, 40]. The primary strategy of most GNNs is to capture underlying message-passing patterns for graph representation. Several effective neural network architectures have been proposed, such as Graph Attention Network (GAT) [2], Graph Convolution Network (GCN) [1], and GraphSAGE [3]. Recently, there has been a surge of interest in exploring transformer-based encoders for graph machine learning [41, 42, 31, 32]. However, a notable limitation of GNNs is their generalization capability. Typically, GNNs are trained on specific tasks within particular datasets, and when faced with new datasets or tasks, they often struggle to consistently perform well across different datasets or downstream tasks [4].

4.2 Self-supervised learning and prompt-tuning for GNNs

To alleviate the demand for labeled data and enhance the robustness of graph models, self-supervised learning is commonly employed in GNN training [43, 22, 44]. Methods like Deep Graph Infomax (DGI) [5] utilize mutual information maximization for pre-training. Other approaches, such as GraphCL [6], GCA [45], GCC [46], and JOAO [47], learn node representations by contrasting positive and negative samples. GraphMAE [48, 49], on the other hand, learns representations by generating samples that resemble the original graph structure. However, these methods typically require fine-tuning the task-specific heads for downstream applications.

Various methods have explored the use of prompt techniques to enhance the generalization of GNNs. To address the inconsistency between pre-training and downstream task objectives, GraphPrompt [7] proposes a unified task template applicable to both stages. Additionally, ProG [8] reformulates various task types into a unified graph-level representation and employs meta-learning techniques to enhance multi-task learning capabilities. However, whether through self-supervised learning or graph prompt methods, fine-tuning is often necessary when handling new datasets. Moreover, when confronted with datasets containing varying numbers of categories, retraining of task heads is required to achieve optimal performance.

4.3 Large language models for graphs

With the rapid advancement of Large Language Models (LLMs) and their remarkable generalization capabilities, leveraging LLMs to address transferability issues in graph machine learning has garnered significant attention [10, 50]. Some methods represent graph structure information as text input to LLMs [9, 11, 12]; however, this approach often leads to suboptimal solutions [13]. Another paradigm involves using LLMs as enhancers [14, 15, 16, 17, 18], where they generate data or node text representations. Despite this, since GNNs are ultimately used for prediction, this approach significantly limits the model's transferability. Recently, considerable efforts have been made to utilize LLMs as predictors. For instance, GraphGPT [19] attempts to align LLMs with pre-trained Graph Transformer encoders through two-stage fine-tuning. However, the fine-tuning, conducted on specific datasets, might weaken the method's transferability. In light of this, LLaGA [20] introduced a novel encoding method that directly translates graph data into sequences compatible with LLMs. However, this approach may compromise performance due to the lack of GNN filtering and aggregation of graph information. Inspired by these challenges, we propose a pre-training strategy that enhances GNN transferability by aligning its representations with the token embeddings of LLMs, resulting in improved performance in zero-shot tasks. Notably, similar to our method, TEST [51] aligns time series representations with several selected LLM token embeddings. However, our approach differs in that we project graph representations into a feature space defined by the principal components of LLM token embeddings. This enables the LLM to function as a zero-shot learner for graph machine learning tasks, rather than just enhancing performance on specific, seen tasks.

5 Limitations

While our TEA-GLM framework demonstrates considerable promise in enhancing zero-shot learning for graph-based tasks, it does have some limitations. Although the framework we designed can be easily applied to graph-level tasks, we have not yet explored the model's performance through specific experiments. This will be addressed in our future work.

6 Conclusion

This paper introduces TEA-GLM, a framework that enhances zero-shot learning in graph machine learning by aligning GNN representations with LLM token embeddings. TEA-GLM uses a linear projector to map graph representations into graph token embeddings and incorporates a unified instruction design to handle various graph tasks at different levels. This approach enables consistent performance across various datasets and tasks without task-specific fine-tuning. Extensive experiments show that TEA-GLM outperforms state-of-the-art methods in accuracy and generalization, demonstrating its effectiveness and efficiency in zero-shot learning for graph tasks.

7 Acknowledgement

This work was supported by the National Key R&D Program of China (2023YFC3304700). Dr. Junjie Wu's work was partially supported by the National Natural Science Foundation of China (72242101, 72031001) and Outstanding Young Scientist Program of Beijing Universities (JWZQ20240201002).

References

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[2] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[3] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.

[4] Mingxuan Ju, Tong Zhao, Qianlong Wen, Wenhao Yu, Neil Shah, Yanfang Ye, and Chuxu Zhang. Multi-task self-supervised graph neural networks enable stronger task generalization. In *The Eleventh International Conference on Learning Representations*, 2023.

[5] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[6] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, pages 28877–28888, 2021.

[7] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. Graphprompt: Unifying pre-training and downstream tasks for graph neural networks. In *Proceedings of the ACM Web Conference 2023*, page 417–428, 2023.

[8] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in one: Multi-task prompting for graph neural networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 2120–2131, 2023.

[9] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, and Jiliang Tang. Exploring the potential of large language models (LLMs) in learning on graph. In *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023.

[10] Jiayan Guo, Lun Du, and Hengyu Liu. Gpt4graph: Can large language models understand graph structured data ? an empirical evaluation and benchmarking. *ArXiv*, abs/2305.15066, 2023.

[11] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[12] Chang Liu and Bo Wu. Evaluating large language models on graphs: Performance insights and comparative analysis. *arXiv preprint arXiv:2308.11224*, 2023.

[13] Jin Huang, Xingjian Zhang, Qiaozhu Mei, and Jiaqi Ma. Can llms effectively leverage graph structural information: when and why. *arXiv preprint arXiv:2309.16595*, 2023.

[14] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Natural language is all a graph needs. *arXiv preprint arXiv:2308.07134*, 2023.

[15] Jianxiang Yu, Yuxiang Ren, Chenghua Gong, Jiaqi Tan, Xiang Li, and Xuecang Zhang. Empower text-attributed graphs learning with large language models (llms). *arXiv preprint arXiv:2310.09872*, 2023.

[16] Lianghao Xia, Ben Kao, and Chao Huang. Opengraph: Towards open graph foundation models. *arXiv preprint arXiv:2403.01121*, 2024.

[17] Zhikai Chen, Haitao Mao, Hongzhi Wen, Haoyu Han, Wei Jin, Haiyang Zhang, Hui Liu, and Jiliang Tang. Label-free node classification on graphs with large language models (LLMs). In *The Twelfth International Conference on Learning Representations*, 2024.

[18] Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *The Twelfth International Conference on Learning Representations*, 2024.

[19] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. *arXiv preprint arXiv:2310.13023*, 2023.

[20] Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. In *ICML*, 2024.

[21] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

[22] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[24] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023.

[25] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, pages 22118–22133, 2020.

[26] Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. Harnessing explanations: LLM-to-LM interpreter for enhanced text-attributed graph representation learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[27] Zhihao Wen and Yuan Fang. Augmenting low-resource text classification with graph-grounded pre-training and prompting. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 506–516, 2023.

[28] Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, Weiwei Deng, Qi Zhang, Lichao Sun, Xing Xie, and Senzhang Wang. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[29] Chenxiao Yang, Qitian Wu, and Junchi Yan. Geometric knowledge distillation: Topology compression for graph neural networks. In *Advances in Neural Information Processing Systems*, 2022.

[30] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old MLPs new tricks via distillation. In *International Conference on Learning Representations*, 2022.

[31] Qitian Wu, Wentao Zhao, Zenan Li, David Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[32] Qitian Wu, Chenxiao Yang, Wentao Zhao, Yixuan He, David Wipf, and Junchi Yan. DIFFormer: Scalable (graph) transformers induced by energy constrained diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.

[33] Jiashun Cheng, Man Li, Jia Li, and Fugee Tsung. Wiener graph deconvolutional network improves graph self-supervised learning. In *AAAI*, 2023.

[34] Jia Li, Zhichao Han, Hong Cheng, Jiao Su, Pengyun Wang, Jianfeng Zhang, and Lujia Pan. Predicting path failure in time-evolving graphs. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1279–1289, 2019.

[35] Xiangguo Sun, Hongzhi Yin, Bo Liu, Hongxu Chen, Qing Meng, Wang Han, and Jiuxin Cao. Multi-level hyperedge distillation for social linking prediction on sparsely observed networks. In *Proceedings of the Web Conference 2021*, page 2934–2945, 2021.

[36] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[37] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1416–1424, 2018.

[38] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 257–266, 2019.

[39] Y. You, T. Chen, Z. Wang, and Y. Shen. L2-gcn: Layer-wise and learned efficient training of graph convolutional networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2124–2132, 2020.

[40] Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018.

[41] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. In *Advances in Neural Information Processing Systems*, 2019.

[42] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, pages 28877–28888, 2021.

[43] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z. Li. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of the ACM Web Conference 2022*, page 1070–1079, 2022.

[44] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[45] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, page 2069–2080, 2021.

[46] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1150–1160, 2020.

[47] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *ICML*, 2021.

[48] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 594–604, 2022.

[49] Zhenyu Hou, Yufei He, Yukuo Cen, Xiao Liu, Yuxiao Dong, Evgeny Kharlamov, and Jie Tang. Graphmae2: A decoding-enhanced masked self-supervised graph learner. In *Proceedings of the ACM Web Conference 2023*, page 737–746, 2023.

[50] Yufei He and Bryan Hooi. Unigraph: Learning a cross-domain graph foundation model from natural language. *arXiv preprint arXiv:2402.13630*, 2024.

[51] Chenxi Sun, Hongyan Li, Yaliang Li, and Shenda Hong. TEST: Text prototype aligned embedding to activate LLM's ability for time series. In *The Twelfth International Conference on Learning Representations*, 2024.

[52] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[53] Mengmei Zhang, Mingwei Sun, Peng Wang, Shen Fan, Yanhu Mo, Xiaoxiao Xu, Hong Liu, Cheng Yang, and Chuan Shi. Graphtranslator: Aligning graph model to large language model for open-ended tasks. In *Proceedings of the ACM Web Conference 2023*, 2024.

A Dataset description

Table 3: Dataset statistics

Domain	Dataset	#Nodes	#Edges	#Classes
Citation	Arxiv	169,343	1,166,243	40
	Pubmed	19,717	44,338	3
	Cora	25,120	91,140	70
E-commerce	Ele-Computer	87,229	721,081	10
	Ele-Photo	48,362	500,928	12
	Book-Children	76,875	1,554,578	24
	Book-History	41,551	358,574	12
	Sports-Fitness	173,055	1,773,500	13

Citation datasets The Arxiv dataset [25] represents a directed citation network among Computer Science (CS) papers from the arXiv preprint server. Each node in this graph corresponds to a paper, while edges represent citation links. The PubMed dataset [26] comprises 19,717 scientific publications from the PubMed database related to diabetes, which are categorized into three distinct classes: Experimentally induced diabetes, Type 1 diabetes, and Type 2 diabetes. This classification reflects the focus of each publication within the broader context of diabetes research. Lastly, the Cora dataset [27], formally known as the "Cora Research Paper Classification Dataset", provides a comprehensive network for analyzing research paper classifications in machine learning. It is an extended version of the dataset commonly referred to in other studies [52], featuring detailed categorizations.

E-commmerce datasets All e-commerce datasets are provided in the TAG benchmark [28]. The Books-Children and Books-History datasets are extracted from the Amazon-Books dataset. Books-Children includes items with the second-level label "Children", while Books-History includes items with the second-level label "History". Each dataset's label corresponds to the three-level label of the book. The Ele-Computers dataset comprises items with the second-level label "Computers", and Ele-Photo includes items with the second-level label "Photo". Each of these datasets is labeled at the third level for electronic products. The Sports-Fitness dataset, sourced from the Amazon-Sports dataset, contains items with the second-level label "Fitness". Nodes in this dataset represent fitness-related items, and an edge between two items indicates they are frequently co-purchased or co-viewed.

B More experimental results

B.1 Legality rate

Table 4: Legality rate of LLM-backbone model (The worst results are marked in gray)

Dataset	Arxiv	Computer	Pubmed	Cora	Children	History	Photo	Sports
Model	Legality rate(%)							
Vicuna-7B-v1.5	99.3	96.7	100.0	95.8	99.2	98.9	94.1	99.6
LLaGA	100.0	100.0	98.9	79.9	93.1	92.4	77.8	94.3
TEA-GLM	100.0	100.0	100.0	92.6	97.0	99.6	99.2	98.5

After training on specific datasets or tasks, large language models (LLMs) may produce invalid or incorrect answers to given questions. For instance, when handling unseen datasets or tasks, LLMs may generate responses that fall outside the set of acceptable answer candidates. To evaluate the impact of the training process on LLM performance, we follow the approach in [53] and use the legality rate to measure the proportion of valid answers produced by the model.

Table 4 demonstrates that the illegality rate of the LLaGA model significantly increases when exposed to datasets it has not previously encountered, suggesting a substantial impact of training methodologies on both the acquisition of knowledge and the model's ability to generalize. Conversely, our model exhibits a notably stable performance across diverse unseen datasets, achieving higher legality rates in several cases.

B.2 F1 score on node classification task

Table 5: Macro F1 of node classification task (**bold** highlights the best result across all methods, while underline highlights the second-best results)

Model type	Model	Citation		E-commerce			
		Pubmed	**Cora**	**Children**	**History**	**Photo**	**Sports**
	MLP	0.246±0.042	0.009±0.004	0.007±0.007	0.023±0.008	0.041±0.023	0.019±0.005
GNN as predictor	GCN	0.187±0.021	0.007±0.001	0.006±0.004	0.024±0.013	0.034±0.007	0.017±0.009
	GraphSAGE	0.257±0.084	0.007±0.003	0.005±0.003	0.029±0.024	0.020±0.011	0.021±0.004
	GAT	0.259±0.065	0.006±0.001	0.063±0.067	0.159±0.117	0.036±0.035	0.091±0.090
	DGI	0.213±0.127	0.004±0.002	0.012±0.004	0.038±0.015	0.045±0.015	0.018±0.005
	GKD	0.247±0.039	0.004±0.001	0.028±0.003	0.060±0.008	0.049±0.015	0.050±0.008
	GLNN	0.221±0.033	0.006±0.001	0.021±0.003	0.064±0.007	0.057±0.002	0.052±0.003
	NodeFormer	0.232±0.089	0.008±0.003	0.019±0.008	0.046±0.031	0.055±0.006	0.049±0.009
	DIFFormer	0.187±0.007	0.007±0.002	0.002±0.002	0.050±0.019	0.069±0.010	0.045±0.007
	OFA	0.287±0.059	0.091±0.013	0.017±0.010	0.026±0.007	0.103±0.007	0.043±0.021
LLM as predictor	Vicuna-7B-v1.5	0.629±0.024	0.109±0.002	**0.279±0.002**	0.349±0.003	0.383±0.001	0.410±0.002
	GraphGPT-std	0.649	0.082	-	-	-	-
	GraphGPT-cot	0.482	0.127	-	-	-	-
	LLaGA	0.778±0.056	0.108±0.014	0.163±0.029	0.144±0.025	0.362±0.039	**0.446±0.035**
	TEA-GLM	**0.839±0.012**	**0.148±0.015**	0.252±0.005	**0.365±0.011**	**0.421±0.032**	0.430±0.009

Due to the absence of a metric to calculate the F1 score while considering the illegality rate, we adopt the methodology used in [53]. For the LLM-backbone models, we only calculate the Macro F1 score for legally permissible responses provided by the model. This calculation method may not accurately reflect the model's performance fully. Therefore, we also report the illegality rate in Table 4. Please note that the accuracy metric is unaffected by illegal responses, which are considered error responses.

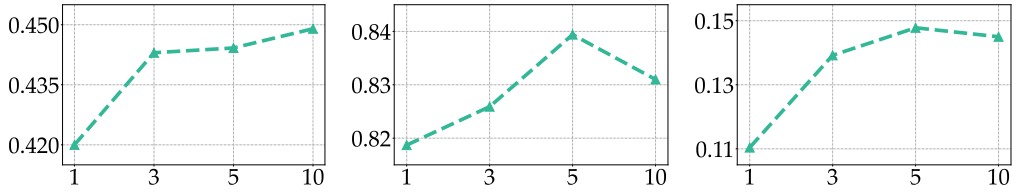
B.3 Supervised results

Table 6: Accuracy and macro F1 on training datasets (**bold** highlights the best result across all methods, while underline highlights the second-best results)

Model type	Model	Arxiv		Computer	
		Acc	F1	Acc	F1
	MLP	0.546±0.004	0.295±0.007	0.420±0.006	0.267±0.005
GNN as predictor	GCN	0.545±0.005	0.317±0.006	0.424±0.012	0.386±0.014
	GraphSAGE	0.556±0.006	0.315±0.008	0.534±0.037	0.347±0.036
	GAT	0.561±0.003	0.339±0.005	0.609±0.035	0.598±0.039
	DGI	0.342±0.024	0.336±0.011	0.594±0.004	0.452±0.008
	GKD	0.393±0.085	0.164±0.029	0.351±0.031	0.155±0.016
	GLNN	0.602±0.004	0.362±0.008	0.393±0.005	0.243±0.007
	NodeFormer	0.544±0.016	0.297±0.029	0.434±0.012	0.288±0.012
	DIFFormer	0.616±0.025	0.356±0.024	0.629±0.012	0.467±0.022
	OFA	0.682±0.006	0.495±0.006	**0.753±0.004**	**0.687±0.006**
LLM as predictor	Vicuna-7B-v1.5	0.347±0.000	0.164±0.001	0.372±0.010	0.304±0.002
	GraphGPT-std	0.626	0.262	-	-
	GraphGPT-cot	0.576	0.228	-	-
	LLaGA	**0.749±0.001**	**0.575±0.003**	0.642±0.004	0.562±0.001
	TEA-GLM	0.655±0.001	0.445±0.002	0.578±0.002	0.496±0.010

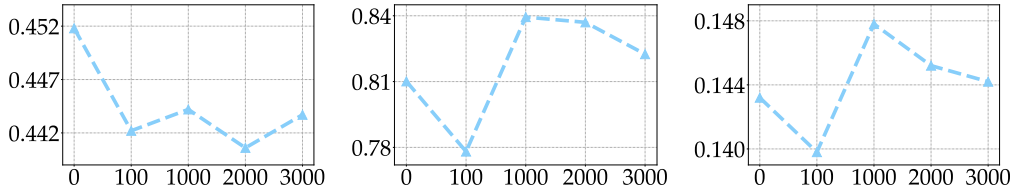
We report the supervised learning results in Table 6. The GNN-backbone models continue to demonstrate robust performance in fitting training data. Similarly, the LLaGA model shows its efficacy in supervised learning scenarios. However, despite their strong performance on training datasets, these models exhibit limited generalization capabilities on unseen datasets as shown in Table 1 and Table 5.

C Parameter sensitivity analysis



(a) Arxiv (b) Pubmed (c) Cora

Figure 3: Impact of number of graph token embeddings (Macro F1)



(a) Arxiv (b) Pubmed (c) Cora

Figure 4: Impact of number of principal components (Macro F1)

Number of graph token embeddings To discuss the impact of the number of graph token embeddings, we set $K \in \{1, 3, 5, 10\}$ and report the results on node classification task in Figure 3. In the context of training datasets and unseen datasets, we observe two distinct patterns. With an increase in the number of graph token embeddings in the training dataset, there is a slight improvement in the model's performance on that dataset. This suggests that in a supervised learning scenario, enhancing the model's performance can be achieved by increasing the quantity of graph token embeddings. Conversely, for unseen datasets, our model requires only a minimal number of graph token embeddings to achieve satisfactory performance, indicating that the number of learnable parameters in our model is significantly less than concurrent works.

Number of principal components We define $P \in \{0, 100, 1000, 2000, 3000\}$ and discuss the results of the node classification task in Figure 4. In supervised learning scenarios, omitting contrastive learning with principal components can lead to a slight increase in accuracy. However, this often makes the model more prone to overfitting on training datasets. When the number of principal components is too small, it adversely affects the model's learning capability. Remarkably, when $P = 1000$, the model demonstrates satisfactory performance. At this level, the principal components capture 50% of the variance of LLM's token embeddings.

D Complete instructions

Given the representation of a paper: **<Token 1> <Token 2> ... <Token K>,** with the following information:
Title: {title}.
Question: Which arXiv CS sub-category does this paper belong to? Please directly give the most likely answer from the following sub-categories: {answer candidates}.

Node Classification

Given the representation of two papers: **<Token 1> <Token 2> ... <Token K>,** with the following information:
Title: First Paper: {title_1}. Second Paper: {title_2}.
Question: What kind of relationship exists between these two papers? Please choose the most likely answer from the following options: "These two papers have citation relationships" or "These two papers may not have citation relationships".

Link Prediction

Figure 5: Instructions for node classification and link prediction

In node classification tasks, we provide candidate labels to facilitate the model's learning process, focusing on discovering the correct answers rather than merely memorizing them. For link prediction, we structure the instructions in a format similar to that of node classification. This approach is designed to enhance the model's ability to transfer learned knowledge effectively across different tasks.

E Cross-task zero-shot results with different pooling methods

Table 7: AUC of link prediction (Cross-task) with different pooling methods

Model	**Citation**		
	Arxiv	**Pubmed**	**Cora**
OFA	0.469	0.481	0.492
Vicuna-7B-v1.5	0.513	0.543	0.527
Vicuna-7B-SPT	0.537	0.535	0.565
GraphGPT-std	0.649	0.501	0.520
LLaGA	0.570	0.569	0.537
TEA-GLM (max)	0.639	0.650	0.566
TEA-GLM (sum)	0.657	0.689	0.586
TEA-GLM (mean)	**0.659**	**0.690**	**0.588**

Considering that different pooling methods may impact cross-task performance, we conducted experiments using three common pooling methods separately, and the results are shown in the Table 7.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We verify the contributions of our proposed method through experiments, and the results in the Sec. 3 effectively demonstrate the contributions we outlined in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the current shortcomings of our method and future research directions in Sec.5

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We disclose all the information needed to reproduce the main experimental results of the paper. In Sec. 2, we talk about the method used in our model. In Sec. 2.3.3 and Sec. 3.1, we give the training process and the detailed settings of our model. We ensure that all the results in our paper can be reproduced.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use publicly available datasets, and we will provide a anonymized link to the code repository.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We discuss the data splits, hyperparameters, random seeds, baselines and type of optimizer in Sec. 3.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: To ensure comparability among different methods, identical data splits were applied to all models. To ensure result robustness, we conduct five experiments with random seed values ranging from 0 to 4, and report the mean and standard deviation of the results. The results are reported in Sec. 3.2 and Appendix B.2

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the computer resources in Sec. 3.1. There is no experiment that requires more than what we mentioned in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research conforms to the ethical guidelines outlined in the NeurIPS Code of Ethics. We have ensured that all aspects of our research, including data collection, data usage, and experimentation, adhere to these standards.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: Our research proposes an improved method that allows large language models to more effectively utilize graph information. We consider this work to be primarily technical research that has not yet been applied to specific real-world scenarios. Therefore, we believe it does not directly produce social impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We doesn't release data or models that have a high risk. This paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: For the models and data we use in our paper, we all cite the original paper. Meanwhile, we provide the license of the data we use in our paper. Arxiv [25] with "ODC-BY". Pubmed [26], Cora [27] and e-commerce datasets in TAG benchmark [28] with "MIT".

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.